# Firma Digitale

## Crittografia

Luciano Margara

Unibo

2022

# Introduzione

In origine i metodi crittografici sono stati sviluppati per garantire la confidenzialità delle comunicazioni, in particolare tra coppie di persone in ambienti ristretti. Con la diffusione delle reti, e in particolare di Internet, il problema si è però enormemente esteso ed ha assunto connotazioni nuove. Sono così emerse tre funzionalità importanti che sono oggi richieste ai protocolli crittografici a seconda dell'uso e del livello di protezione desiderato.

# Identificazione 1

Identificazione: un sistema di elaborazione deve essere in grado di accertare l'identità di un utente che richiede di accedere ai suoi servizi.

# Autenticazione 2

Autenticazione: il destinatario di un messaggio deve essere in grado di accertare l'identità del mittente e l'integrità del crittogramma ricevuto, cioè che questo non sia stato modificato o sostituito nella trasmissione. Deve quindi essere difficile a un intruso spacciarsi per un altro utente o modificare i messaggi da questo inviati.

# Firma digitale 3

Firma digitale: entra in gioco quando il mittente e il destinatario non si fidano l'uno dell'altro. La firma digitale deve possedere i seguenti tre requisiti:

1. il mittente non deve poter negare di aver inviato un messaggio da lui firmato
2. il destinatario deve poter accertare l'identità del mittente e l'integrità del messaggio ricevuto (cioè deve poter autenticare il messaggio)
3. il destinatario non deve poter sostenere di aver ricevuto un messaggio diverso da quello inviatogli dal mittente.

# Funzioni hash classiche

Le funzioni hash classiche sono funzioni che trasformano un insieme di dati di grandi dimensioni X (dominio) in un insieme di dimensioni più piccole Y (codominio).

Una funzione hash $f : X \to Y$ è definita per un dominio $X$ e un codominio $Y$ finiti e tali che $n = |X| >> m = |Y|$. Nella costruzione di algoritmi queste funzioni sono impiegate per operare su elementi $x \in X$ attraverso la loro immagine (o fingerprint) $y = f(x) \in Y$, essenzialmente per due scopi. Anzitutto la rappresentazione di $y$ richiede $\log_2(m)$ bit ed è quindi più breve di quella di $x$ che ne richiede $\log_2(n)$. Inoltre $Y$ può avere un "struttura" assente in $X$: per esempio gli elementi $y$ corrispondono alle posizioni di un vettore di $m$ celle in cui memorizzare informazioni associate a $m$ diversi elementi di $X$.

l'insieme dei valori hash Y può essere organizzato in un modo che semplifica operazioni come la memorizzazione e la ricerca, mentre l'insieme originale X potrebbe non avere una struttura altrettanto utile per questi scopi.

# Funzioni hash classiche

Partizione = collezione di sottoinsiemi disgiunti

La differenza di cardinalità tra i due insiemi $X$ e $Y$ implica che esiste una partizione di $X$ in sottoinsiemi disgiunti $X_1, \ldots, X_m$ tali che, per ogni valore dell'indice $i$, tutti gli elementi in $X_i$ hanno come immagine uno stesso elemento di $Y$ . Una buona funzione hash classica deve assicurare che i sottoinsiemi $X_1, \ldots, X_m$ abbiano circa la stessa cardinalità, in modo che due elementi $x_h, x_k$ estratti a caso da $X$ abbiano probabilità circa pari a $1/m$ di avere la stessa immagine in $Y$. (quindi si cerca di ridurre il numero di collisioni)

# Funzioni hash classiche

La funzione hash deve garantire che elementi molto simili di X appartengano a sottoinsiemi diversi. Questo significa che, se due elementi di X sono simili, non devono necessariamente essere mappati allo stesso valore in Y. In altre parole, la funzione hash deve distribuire uniformemente gli elementi di X in Y, evitando di creare raggruppamenti basati sulla somiglianza di X.

Inoltre deve assicurare che elementi di $X$ molto simili tra loro appartengano a due sottoinsiemi diversi.

Naturalmente se si lavora su molti elementi di $X$ alcuni avranno inevitabilmente la stessa immagine (collisione) e l'algoritmo che impiega la funzione hash $f$ dovrà gestire questa situazione.

# Funzioni hash one-way o crittografiche

Le funzioni hash one-way devono assicurare proprietà aggiuntive a quelle classiche.

1. Per ogni elemento $x \in X$ è computazionalmente facile calcolare $f(x)$
2. Per la maggior parte degli elementi $y \in Y$ è computazionalmente difficile determinare un $x$ tale che $f(x) = y$
3. Risulta computazionalmente difficile determinare una coppia di elementi $x', x''$ in $X$ tali che $f(x') = f(x'')$

riducendo il rischio di collisioni intenzionali

# Funzioni hash one-way o crittografiche

Tutte le funzioni hash dovrebbero soddisfare la proprietà (1). La proprietà (2) (detta one-way) e la proprietà (3) (detta claw-free), necessarie per impieghi crittografici, dovrebbero essere garantite attraverso una dimostrazione formale di intrattabilità. Poiché però le funzioni che consentono una tale dimostrazione sono difficili da utilizzare ci si accontenta di funzioni semplici per cui siano falliti in pratica tutti i tentativi di attacco.

Idealmente, le funzioni hash crittografiche dovrebbero essere accompagnate da una dimostrazione formale di intrattabilità, ovvero una prova matematica che ne garantisca la sicurezza. Tuttavia, tali funzioni sono spesso complesse e poco pratiche. Nella realtà, si preferisce usare funzioni più semplici ma che siano state testate contro attacchi, e per le quali non siano stati trovati metodi pratici di violazione.

# Famiglia *MD*5

MD5 (Message Digest versione 5) costruisce un'immagine di 128 bit per qualunque messaggio. Fu proposta da Rivest nel 1992 ed è stata usata per molto tempo in applicazioni crittografiche. Dieci anni più tardi sono stati però individuati diversi attacchi, in particolare legati alla imperfetta proprietà di claw-free della funzione, che ne sconsigliano l'impiego in molte applicazioni crittografiche. Data la sua semplicità ed efficienza MD5 rimane però largamente usata in particolare fuori dalla crittografia, per esempio per la ricerca e il confronto di file su Internet.

# Famiglia *RIPEMD* – 160

Una successiva funzione in competizione con essa è RIPEMD-160 (Race Integrity Primitive Evaluation Message Digest), nata nel 1995 nell'ambito di un progetto dell'Unione Europea, che produce immagini di 160 bit ed è esente dai difetti di MD5. La struttura di queste funzioni non è molto diversa da quelle della famiglia SHA e MD5

# Famiglia *SHA*

La prima funzione SHA (Secure Hash Algorithm), nota in seguito come SHA0, fu proposta dal NIST nel 1993 in competizione a MD5 e fu presto ritirata a causa di un punto di debolezza scoperto al suo interno. La National Security Agency (NSA) la sostituì con la nuova versione SHA1 che come la precedente opera su sequenze lunghe fino a $2^{64} - 1$ bit e produce immagini di 160 bit.

# Famiglia *SHA*

SHA1 è largamente usata in molti protocolli crittografici anche se, a causa di possibili attacchi di importanza forse più teorica che pratica, non è più certificata come standard dal 2010. Seguirono così diverse funzioni, anch'esse progettate da NSA, che operano su blocchi di dimensioni diverse a seconda della funzione scelta (ad esempio SHA256 e SHA512).

- SHA-256: Produce un hash di 256 bit ed opera su blocchi di 512 bit.
- SHA-512: Produce un hash di 512 bit ed opera su blocchi di 1024 bit.

# Identificazione

Consideriamo l'accesso di un utente alla propria casella di posta elettronica, o ai file personali memorizzati su un calcolatore ad accesso riservato ai membri della sua organizzazione. L'utente solitamente inizia il collegamento inserendo un codice di login (per esempio nome o cognome o e-mail) e successivamente una password.

# Password buone e password banali

Gli utenti sprovveduti si servono di password banali per paura di dimenticarle, mentre i più attenti scelgono password molto strane per rendere difficile una loro individuazione: non ci preoccuperemo dei primi perché alla loro disattenzione è comunque difficile porre rimedio, ma ci interesseremo degli utenti esperti che desiderano essere protetti efficacemente da attacchi esterni o interni al sistema a cui si connettono.

# Canale protetto

Assumiamo che il canale attraverso il quale avviene il processo di identificazione sia protetto in lettura e scrittura. L'attacco può essere sferrato soltanto da un utente locale, per esempio dal suo amministratore (o super-user) che ha accesso a tutti i file memorizzati e in particolare a quello contenente le password degli utenti oppure da un hacker che è riuscito a intrufolarsi nel sistema.

# Canale protetto

Il meccanismo di identificazione deve prevedere una opportuna cifratura delle password, che può essere realizzata in modo semplice ed efficace impiegando funzioni hash one-way. Il metodo che ora descriviamo è utilizzato per esempio nei sistemi UNIX.

# Canale protetto

Quando un utente $U$ fornisce per la prima volta la password, il sistema associa a $U$ due sequenze binarie $S, Q$ che memorizza nel file delle password in luogo di questa. $S$ (seme) è prodotta da un generatore pseudo-casuale. $Q$ è l'immagine, secondo una funzione hash one-way, della sequenza ottenuta concatenando $S$ alla password di $U$

# Canale protetto

A ogni successiva connessione dell'utente il sistema recupera il seme $S$ dal file delle password, lo concatena con la password fornita da $U$ e calcola l'immagine della nuova sequenza: se questa coincide con $Q$ l'identificazione ha successo. Si noti che l'utilizzo delle funzioni hash è sempre "in avanti", mai "all'indietro".

# Canale protetto

La presenza di un seme casuale diverso per ogni utente impedisce al crittoanalista di scoprire se due utenti hanno scelto due password uguali guardando solo le loro immagini, e rende inoltre impossibile un attacco simultaneo alle chiavi su tutto il file.

# Canale non protetto

L’assunzione sulla protezione del canale risulta ovviamente molto forte, e comunque inaccettabile in un contesto distribuito come quello di Internet. Se il canale è insicuro la password può essere intercettata durante la sua trasmissione in chiaro: è quindi opportuno che il sistema non maneggi mai direttamente la password ma una sua immagine inattaccabile.

In sintesi, il sistema deve evitare di maneggiare direttamente le password in chiaro, affidandosi invece a trasformazioni crittografiche (hash crittografiche) che rendano impraticabile qualsiasi tentativo di recupero delle informazioni sensibili anche in caso di intercettazioni.

# Canale non protetto: identificazione con RSA

encode decode

Siano $(e, n)$ e $(d)$ le chiavi pubblica e privata di un utente $U$ che vuole accedere ai servizi offerti da un sistema S.

1. $U$ richiede accesso a $S$
2. $S$ genera un numero casuale $r < n$ e lo invia a $U$.
3. $U$ calcola $f = r^d \mod n$ con la sua chiave privata $(d)$, e spedisce $f$ a $S$. Questo valore prende il nome di firma di $U$ su $r$.
4. $S$ usa la chiave pubblica $(e, n)$ di $U$, calcola $r' = f^e \mod n$ e controlla che $r = r'$.

# Canale non protetto: identificazione con RSA

Si noti l'inversione d'ordine delle operazioni di cifratura e decifrazione rispetto all'impiego standard del cifrario RSA: ciò è possibile perché le due funzioni sono commutative, ovvero

$$(m^e \bmod n)^d \bmod n = (m^d \bmod n)^e \bmod n$$

# Canale non protetto: identificazione con RSA

Il valore $f$ può essere generato solo da $U$ che possiede la chiave privata $(d)$: quindi se la verifica va a buon fine il sistema ha la garanzia che l'utente che ha richiesto l'identificazione sia effettivamente $U$, anche se il canale è insicuro.

# Canale non protetto: identificazione con RSA

L'identificazione con la chiave pubblica si presta a un attacco molto subdolo. $S$ chiede a $U$ di applicare la sua chiave segreta a una sequenza $r$ che S stesso ha generato. Un sistema disonesto potrebbe inviare una $r$ scelta ad-hoc per ricavare qualche informazione sulla chiave privata di $U$ attraverso l'esame della trasformazione $f$ di $r$ costruita con tale chiave, soprattutto nel caso che $U$ debba identificarsi molte volte con $S$ e questo possa quindi impiegare una serie mirata di scelte di $r$. Questo tipo di attacco può permettere a $U$ di ridurre notevolmente il campo di variabilità della chiave privata di $U$.

# Autenticazione: scenario

Il processo di identificazione è volto a stabilire l'identità di un utente ma non si occupa dei suoi messaggi.
Consideriamo allora di nuovo la situazione fondamentale in cui due partner *Mitt* e *Dest* devono scambiarsi un messaggio $m$ in maniera confidenziale, ma *Dest* deve anche autenticare il messaggio accertando l'identità di *Mitt* e l'integrità di $m$.

Vediamo ora come ciò possa realizzarsi impiegando una chiave segreta $k$ concordata a questo scopo tra *Mitt* e *Dest*.

# MAC con funzioni hash one-way

il MAC è un'immagine breve del messaggio che può essere stata generata solo da un mittente conosciuto dal destinatario previ opportuni accordi. Ad esempio utilizzando una chiave segreta e condivisa $k$.

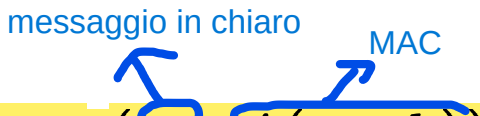
il MAC è uno strumento molto potente. Ne sono state proposte diverse realizzazioni basate su cifrari simmetrici e asimmetrici, nonché su funzioni hash one-way. Quest'ultimo approccio conduce a protocolli molto economici che sono attualmente i più usati in pratica

(asimmetrico)

# MAC con funzioni hash one-way

Si definisce il MAC come $A(m, k) = h(mk)$. Ovvero si applica la funzione hash sulla concatenazione di $m$ e $k$. Risulta così computazionalmente difficile per un crittoanalista scoprire la chiave segreta $k$.

# Autenticazione con chiave segreta




1. *Mitt* spedisce $(m, A(m,k))$ a *Dest*
2. *Dest* ricalcola $A(m,k)$ a partire dal messaggio ricevuto e dalla chiave $k$ condivisa.
3. *Dest* verifica se il MAC ricevuto è uguale al MAC ricalcolato

# Autenticazione con chiave segreta + confidenzialità

Messaggio cifrato

MAC

1. *Mitt* spedisce $(C(m, k'), A(m, k))$ a *Dest*
2. *Dest* decodifica $C(m, k')$ (chiave pubblica o chiave ~~privata~~ segreta) (in base al tipo di Crittografia utilizzata (simmetrica o asimmetrica))
3. *Dest* ricalcola $A(m, k)$ a partire dal messaggio decodificato e dalla chiave $k$ condivisa.
4. *Dest* verifica se il MAC ricevuto è uguale al MAC ricalcolato

# Firma Manuale

Perché si appone una firma manuale ?

La firma manuale è utilizzata comunemente per provare l'autenticità o la paternità di un documento, o per siglare un accordo formulato in esso. Si firmano documenti quando i soggetti coinvolti non si fidano uno dell'altro.

# Firma Manuale: proprietà

1. La firma è autentica e non è falsificabile, dunque costituisce prova che chi l'ha prodotta è veramente colui che ha sottoscritto il documento.
2. La firma non è riutilizzabile, e quindi risulta legata strettamente al documento su cui è stata apposta.
3. Il documento firmato non è alterabile, e quindi chi ha prodotto la firma è sicuro che questa si riferirà solo al documento sottoscritto nella sua forma originale.
4. La firma non può essere ripudiata da chi l'ha apposta, e costituisce dunque prova legale di un accordo o di una dichiarazione contenuta nel documento.

# Firma digitale

La firma digitale deve soddisfare le stesse proprietà firma manuale e possibilmente godere di proprietà aggiuntive.

La firma digitale non può non dipendere dal documento sul quale viene posta. Altrimenti potrebbe essere copiata (essendo digitale) e riutilizzata a piacimento

# Firma digitale

La firma digitale può essere realizzata sia mediante protocolli crittografici simmetrici che asimmetrici. I protocolli simmetrici danno però luogo in genere a realizzazioni più complicate e computazionalmente costose. Per questo utilizzeremo protocolli crittografici asimmetrici

# Messaggio in chiaro e firmato 1

$U$ generico utente, $k_U[prv]$ e $k_U[pub]$ chiavi di $U$, $C$ e $D$ funzioni di cifratura e decifrazione asimmetriche.

*Firma* L'utente $U$ genera la firma $f = D(m, k_U[prv])$ per $m$. Il messaggio firmato può essere spedito a un qualunque altro utente $V$ come tripla $(U, m, f)$.

*Verifica* L'utente $V$ riceve $(U, m, f)$ e verifica l'autenticità della firma $f$ calcolando $C(f, k_U[pub])$ e controllando che questo valore sia uguale a $m$. L'indicazione del mittente (etichetta $U$ nella tripla) consente al destinatario di recuperare la sua chiave pubblica.

# Messaggio in chiaro e firmato 1

Occorre commutatività del protocollo utilizzato ovvero $C(D(m)) = D(C(m)) = m$

- La firma è autentica e non falsificabile poiché la chiave $k_U[prv]$ utilizzata per produrla è nota solo a $U$ e la funzione $D$ è one-way.
- Il documento firmato $(U, m, f)$ non può essere alterato (se non da $U$), pena la perdita di consistenza tra $m$ e $f$. Solo $U$ può aver prodotto $f$ e non la può ripudiare.
- Chiunque può verificare la firma, anche un giudice
- La firma $f$ non è riutilizzabile su un altro documento

# Difetti del protocollo 1

- La firma è lunga quanto il documento firmato
- Il messaggio viaggia in chiaro e a volte questa cosa è indesiderata

# Messaggio cifrato e firmato 2

*Firma* L'utente $U$ genera la firma $f = D(m, k_U[prv])$ per $m$. $U$ calcola poi $c = C(f, k_V[pub])$. $U$ spedisce $(U, c)$ a $V$.

*Verifica* L'utente $V$ riceve $(U, c)$. $V$ decifra $c$ con la sua chiave privata, cioè calcola $D(c, k_V[prv])$ ottenendo un valore pari a $f$. $V$ cifra tale valore con la chiave pubblica di $U$ ottenendo $C(f, k_U[pub]) = m$.

# Messaggio cifrato e firmato: RSA 2

*Firma* L'utente $U$ genera la firma $f = m^{d_U} \mod n_U$ per $m$. $U$ calcola poi $c = f^{e_V} \mod n_V$. $U$ spedisce $(U, c)$ a $V$.

*Verifica* L'utente $V$ riceve $(U, c)$. $V$ decifra $c$ con la sua chiave privata, cioè calcola $c^{d_V} \mod n_V$ ottenendo un valore pari a $f$. $V$ cifra tale valore con la chiave pubblica di $U$ ottenendo $f^{e_U} \mod n_U = m$. Se il valore ottenuto è significativo, $V$ conclude che il messaggio è autentico.

(si riferisce al fatto che, dopo aver effettuato la verifica della firma, V controlla se il messaggio risultante (ottenuto cifrando la firma con la chiave pubblica di U) ha senso nel contesto applicativo)

# Messaggio cifrato e firmato: RSA

La correttezza del procedimento è soggetta a una condizione che deriva dalle proprietà algebriche del cifrario RSA. Deve infatti valere la relazione $n_U \leq n_V$ perché risulti $f < n_V$ e la firma possa essere cifrata correttamente da $U$ e spedita a $V$. Ma poiché ciò impedirebbe a $V$ di inviare messaggi firmati e cifrati a $U$, ogni utente deve stabilire chiavi distinte per la firma e per la cifratura.

Se si imponesse la relazione 'nU<nV' in modo fisso, si verificherebbe un problema nella comunicazione inversa:
- Se V volesse inviare un messaggio firmato e cifrato a U, allora si dovrebbe avere 'nV<nU' per rispettare la stessa condizione (cioè, affinché il valore ottenuto dalla firma di V sia minore di nU e possa essere cifrato con la chiave pubblica di U).
- Queste due condizioni sono in conflitto tra loro se si utilizzasse un'unica coppia di chiavi per ogni utente.

-> Per risolvere il problema, ogni utente deve adottare due coppie di chiavi RSA distinte:
Chiave per la firma e Chiave per la cifratura

# Messaggio cifrato e firmato in hash 3

*Firma* Il mittente $U$ calcola $h(m)$ e genera la firma $f = D(h(m), k_U[prv])$. $U$ calcola separatamente il crittogramma $c = C(m, k_V[pub])$ e spedisce a $V$ la tripla $(U, c, f)$.

*Verifica* Il destinatario $V$ riceve la tripla $(U, c, f)$. $V$ decifra $c$ come $D(c, k_V[prv])$ ottenendo $m$. $V$ calcola separatamente $h(m)$ e $C(f, k_U[pub])$. Se questi due valori sono uguali conclude che il messaggio è autentico.

# Attacco man in-the-middle

I protocolli di firma descritti hanno una debolezza che deriva proprio da uno dei pregi dei cifrari asimmetrici: le chiavi di cifratura sono pubbliche. Una chiave pubblica può infatti essere recuperata dalla pagina Web del mittente oppure attraverso un e-mail: tutto ciò sembra ragionevole, ma non tiene conto del fatto che un crittoanalista attivo può intromettersi proprio in questa fase iniziale del protocollo, pregiudicando il suo corretto svolgimento.

# Attacco man in-the-middle

Il crittoanalista $X$ si intromette nella comunicazione tra un mittente $U$ e un destinatario $V$ e si comporta agli occhi di $U$ come se fosse $V$ e agli occhi di $V$ come se fosse $U$, intercettando e bloccando le comunicazioni che provengono dai due.

# Attacco man in-the-middle

1. $U$ richiede a $V$ la sua chiave pubblica, per esempio attraverso un e-mail.
2. $X$ intercetta la risposta contenente $k_V[pub]$ e la sostituisce con la sua chiave pubblica $k_X[pub]$.
3. Da questo momento in poi $X$ si pone costantemente in ascolto in attesa dei crittogrammi spediti da $U$ a $V$, cifrati mediante $k_X[pub]$. Ciascuno di questi crittogrammi viene rimosso dal canale, decrittato da $X$, cifrato mediante $k_V[pub]$ e rispedito a $V$.

# Certification Authority (CA)

Per evitare attacchi come quello appena descritto sono nate le (Key) Certification Authority (brevemente CA), enti preposti alla certificazione di validità delle chiavi pubbliche. La CA autentica l'associazione (utente $\leftrightarrow$ chiave pubblica ) emettendo un certificato digitale, così come l'anagrafe di un comune autentica l'associazione (dati personali $\leftrightarrow$ fotografia) rilasciando una carta di identità.

# Certificato digitale

Il certificato digitale consiste della chiave pubblica e di una lista di informazioni relative al suo proprietario, opportunamente firmate dalla CA. Per svolgere correttamente il suo ruolo la CA mantiene un archivio di chiavi pubbliche sicuro, accessibile a tutti e protetto da attacchi in scrittura non autorizzati. La chiave pubblica della CA è nota agli utenti che la mantengono protetta da attacchi esterni e la utilizzano per verificare la firma della CA stessa sui certificati.

# Certificato digitale: contenuto

- Una indicazione del suo formato (numero di versione).
- Il nome della CA che l'ha rilasciato.
- Un numero seriale che individua univocamente questo certificato all'interno della CA emittente.
- La specifica dell'algoritmo utilizzato dalla CA per creare la firma elettronica, combinata con una descrizione del formato dei parametri di cui esso ha bisogno.

# Certificato digitale: contenuto

- Il periodo di validità del certificato (data di inizio e data di fine).
- Il nome dell'utente a cui questo certificato si riferisce e una serie di informazioni a esso legate.
- Un'indicazione del protocollo a chiave pubblica adottato da questo utente per la cifratura e la firma: nome dell'algoritmo, suoi parametri e chiave pubblica dell'utente.
- Firma della CA eseguita su tutte le informazioni precedenti.

# Certificato digitale: contenuto

Se un utente $U$ vuole comunicare con un altro utente $V$ può richiedere la chiave pubblica di quest'ultimo alla CA, che risponde inviando a $U$ il certificato digitale di $V$.
Poiché $U$ conosce la chiave pubblica della CA, egli può controllare l'autenticità del certificato verificando il suo periodo di validità e la firma prodotta dalla CA su di esso.
Se questi controlli vanno a buon fine, la chiave pubblica di $V$ contenuta nel certificato è corretta e $U$ la può utilizzare per avviare una comunicazione cifrata con $V$.
Un crittoanalista potrebbe intromettersi soltanto falsificando quella certificazione, ma si assume che la CA sia fidata e il suo archivio delle chiavi sia inattaccabile.

# Messaggio cifrato e firmato in hash e certificato

Firma, cifratura e certificazione.

Il mittente $U$ calcola $h(m)$ e genera la firma $f = D(h(m), k_U[prv])$. $U$ calcola il crittogramma $c = C(m, k_V[pub])$ e spedisce a $V$ la tripla $(cert_U, c, f)$.

Si noti che $cert_U$ contiene la chiave $k_U[pub]$ e la specificazione della funzione $h$ utilizzata.

# Messaggio cifrato e firmato in hash e certificato

Verifica del Certificato.
Il destinatario $V$ riceve la tripla $(cert_U, c, f)$ e verifica l'autenticità di $cert_U$ (e quindi della $k_U[pub]$ ivi contenuta) utilizzando la propria copia della chiave pubblica di CA.

# Messaggio cifrato e firmato in hash e certificato

Decifrazione e verifica della firma.

$V$ decifra il crittogramma $c$ mediante la sua chiave privata ottenendo $m = D(c, k_V[prv])$ verifica l'autenticità della firma apposta da $U$ su $m$ controllando che risulti $C(f, k_U[pub]) = h(m)$.

# Elenco CA (Aprile 2023, fonte: Wikipedia)

| Rank | Issuer | Usage | Market Share |
|---|---|---|---|
| 1 | IdenTrust | 38.5% | 43.6% |
| 2 | DigiCert Group | 13.1% | 14.5% |
| 3 | Sectigo (Comodo Cybersecurity) | 12.1% | 13.4% |
| 4 | GlobalSign | 16.1% | 16.7% |
| 5 | Let's Encrypt | 5.8% | 6.4% |
| 6 | GoDaddy Group | 4.8% | 5.3% |

# Agid


Presidenza del Consiglio dei Ministri
AGID
Agenzia per
l'Italia digitale

# Prestatori servizi fiduciari

## Prestatori di servizi fiduciari attivi in Italia

| Ragione sociale | Indirizzo della sede legale | Rappresentante legale | Man. oper. certificatore | Data iscrizione | Man. oper. sottoscritto AgID |
|---|---|---|---|---|---|
| Actalis S.p.A. | Via S. Clemente, 53 – 24036 Ponte San Pietro (BG) , IT | Cecconi Giorgio | Link | 28/03/2002 | Manuali operativi **Data:** 18/07/2023 |
| Aruba Posta Elettronica Certificata S.p.A. | Via San Clemente n. 53 - 24036 Ponte San Pietro (BG) | Cecconi Giorgio | Link | 06/12/2007 | Manuali operativi **Data:** 18/07/2023 |
| Banca d'Italia | Via Nazionale, 91 - 00184 Roma, IT | il Governatore pro tempore | Link | 23/01/2008 | Manuali operativi **Data:** 12/01/2022 |
| Cedacri S.p.A. (già Cedacrinord S.p.A.) | via del Conventino, 1 - 43044 Collecchio (PR), IT | Corrado Sciolla, Amministratore Delegato | Link | 15/11/2001 | Manuali operativi **Data:** 12/01/2022 |
| Consiglio Nazionale del Notariato | via Flaminia, 160 - 00196 Roma, IT | Il Presidente pro tempore | Link | 12/09/2002 | Manuali operativi **Data:** 12/01/2022 |

# Prestatori servizi fiduciari

| | | | | | |
|---|---|---|---|---|---|
| In.Te.S.A. S.p.A. | Strada Pianezza, 289 - 10151 Torino IT | Andrea Agnello, Amministratore Delegato | Link | 22/03/2001 | Manuali operativi **Data:** 18/07/2023 |
| InfoCamere S.C.p.A | Via G.B. Morgagni 13 -00161 Roma | Lorenzo Tagliavanti | Link | 08/07/2020 | Manuali operativi **Data:** 12/01/2022 |
| InfoCert S.p.A. | Piazza Sallustio, 9 - 00187 Roma, IT | Daniele Vaccarino, Presidente CdA | Link | 19/07/2007 | Manuali operativi **Data:** 27/11/2023 |
| inPoste.it S.p.A. | via Giuseppe Gioachino Belli, 86 - 00193 Roma | Alberto Lenza | Link | 06/02/2024 | Manuali operativiotNotice - SERCQ **Data:** 06/02/2024 |
| Intesa Sanpaolo S.p.A. (già Sanpaolo IMI S.p.A. e Banca Intesa S.p.A.) | P.za San Carlo, 156 - 10126 Torino, IT | Messina Carlo, Consigliere delegato e CEO | Link | 07/04/2004 | Manuali operativi **Data:** 18/07/2023 |
| Intesi Group S.p.A. | Via Torino, 48 - 20123 Milano, IT | Paolo Sironi | Link | 19/01/2018 | Manuali operativi **Data:** 28/02/2022 |

# Prestatori servizi fiduciari

| | | | | | |
|---|---|---|---|---|---|
| Ministero Della Difesa - CORDIFESA (ex Comando C4 Difesa) | Via Stresa, 31/B - 00135 Roma, IT | Gen.Div. Sergio Antonio SCALESE | Link | 20/09/2006 | Manuali operativi **Data:** 29/12/2022 |
| Namirial S.p.A. | Via Caduti sul Lavoro, 4 - 60019 Senigallia (AN), IT | Massimiliano Pellegrini, Amministratore Delegato. | Link | 03/11/2010 | Manuali operativi **Data:** 28/04/2023 |
| NexiPayments S.p.A. | Corso Sempione 55 - 20149 Milano, IT | Michaela Castelli | Link | 01/07/2018 | Manuali operativi **Data:** 22/02/2022 |
| Notartel S.p.A. | Via Flaminia 162, 00196 Roma | Gian Mario Braido | Link | 26/07/2022 | Manuali operativi **Data:** 27/07/2022 |
| Poste Italiane S.p.A. | Viale Europa, 190 - 00144 Roma IT | Matteo Del Fante | Link | 29/03/2017 | Manuali operativi **Data:** 18/07/2023 |
| Register S.p.A. | Viale della Giovine Italia 17 - 50122 Firenze | Claudio Corbetta - Amministratore Delegato | Link | 10/06/2019 | Manuali operativi **Data:** 13/01/2021 |